# **Capturing PerfMon data to a database**

Contents

[**Capturing PerfMon data to a database** 1](#_Toc425238226)

[Contents 1](#_Toc425238227)

[1 – SQL\_PerfMonCollector 1](#_Toc425238228)

[2 – Create Database 1](#_Toc425238229)

[3 – Create Login and User 2](#_Toc425238230)

[4 – Permissions 2](#_Toc425238231)

[5 – Create ODBC data source 3](#_Toc425238232)

[6 – Create Data Collector Set 5](#_Toc425238233)

[7 – Add NonClustered Index and Page Compression 8](#_Toc425238234)

[8 – Remove SQL\_PerfMonCollector 8](#_Toc425238235)

[9 – Create SQL Agent Job 8](#_Toc425238236)

[10 – Purge Data 11](#_Toc425238237)

[11 – Create Views and Functions 12](#_Toc425238238)

The following describes the steps needed to create a PerfMon capture to a database.

### 1 – SQL\_PerfMonCollector

Remote onto server using an account with Local Administrator rights and add SQL\_PerfMonCollector to the Administrators group.

### 2 – Create Database

Using an account with sysadmin rights, create the database, adding the correct filepath locations to the script below (and backup to stop trans log job failing)

CREATE DATABASE [zzPerfMon]

ON PRIMARY

(NAME = N'zzPerfMon', FILENAME = N'zzPerfMon.mdf', SIZE = 5120MB, MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB)

LOG ON

(NAME = N'zzPerfMon\_log', FILENAME = N'zzPerfMon\_log.ldf', SIZE = 1024MB, MAXSIZE = UNLIMITED, FILEGROWTH = 1024MB)

GO

### 3 – Create Login and User

Create Login that captures the PerfMon counters on the SQL Instance, specifying the correct domain.

CREATE LOGIN [<domain>\SQL\_PerfMonCollector] FROM WINDOWS WITH DEFAULT\_DATABASE= [zzDefaultDB], DEFAULT\_LANGUAGE= [us\_english]

GO

USE [zzDefaultDB]

GO

CREATE USER [<domain>\SQL\_PerfMonCollector]

FOR LOGIN [<domain>\SQL\_PerfMonCollector]

GO

USE zzPerfMon

GO

CREATE USER [<domain>\SQL\_PerfMonCollector]

FOR LOGIN [<domain>\SQL\_PerfMonCollector]

GO

/\*delete depending on version\*/

/\*2012\*/ALTER ROLE [db\_owner] ADD MEMBER [<domain>\SQL\_PerfMonCollector]

/\*2008\*/sp\_addrolemember 'db\_owner', [<domain>\SQL\_PerfMonCollector]

GO

### 4 – Permissions

Grant SYSTEM account access to PerfMon capture database

USE [zzPerfMon]

GO

IF NOT EXISTS

(SELECT 1 FROM sys.database\_principals WHERE name = 'NT AUTHORITY\SYSTEM')

BEGIN

CREATE USER [NT AUTHORITY\SYSTEM] FOR LOGIN [NT AUTHORITY\SYSTEM]

END

GO

USE [zzPerfMon]

GO

/\*delete depending on version\*/

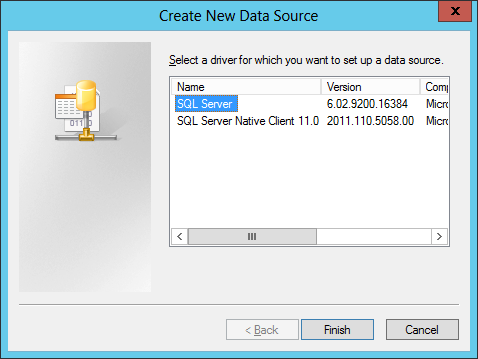
/\*2012\*/ALTER ROLE [db\_owner] ADD MEMBER [NT AUTHORITY\SYSTEM]

/\*2008\*/sp\_addrolemember 'db\_owner', [NT AUTHORITY\SYSTEM]

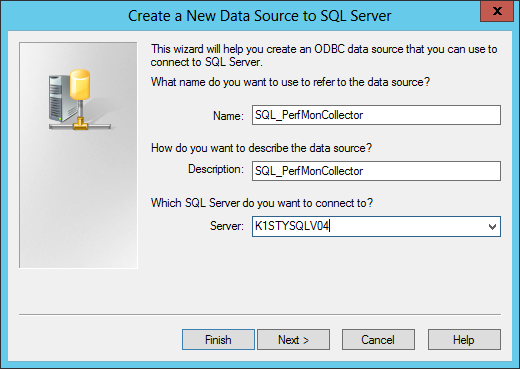
GO

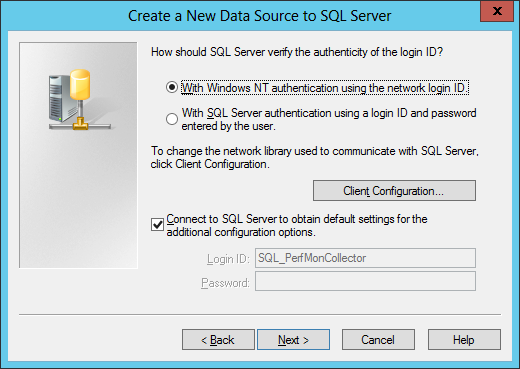
### 5 – Create ODBC data source

Log onto server with account used in step 2. Create 64-bit ODBC data source System DSN (2nd Tab) (under Admin Tools in Windows 2008). Make sure the data source uses the SQL Server driver (rather than Native Client)

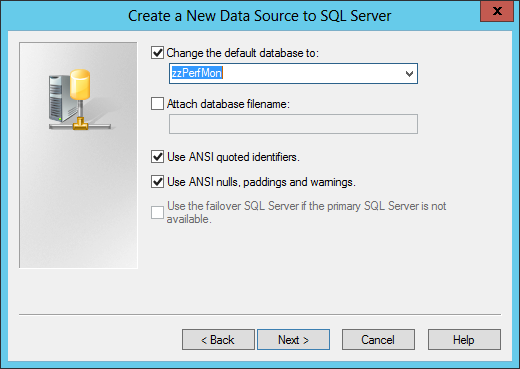


Choose SQL\_PerfMonCollector as the name and the Instance you want to capture the counters from.





Change the database to the same as created in step 1

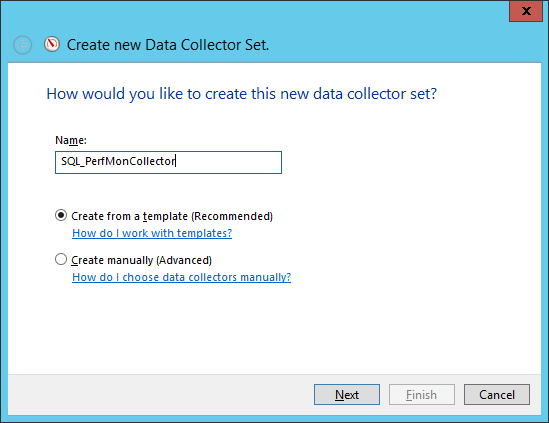


Click Next and Finish

For instances on failover clusters where there are more than one instance contained on the cluster, an ODBC data source needs to be created on each node. The name of each ODBC data source needs to be consistent across all nodes in the cluster and also unique to the instance.

### 6 – Create Data Collector Set

Create the Data Collector Set. In PerfMon, choose Data Collector Sets, right mouse click User Defined and then New Data Collector Set. Set the name to SQL\_PerfMonCollector and choose Create from a template.



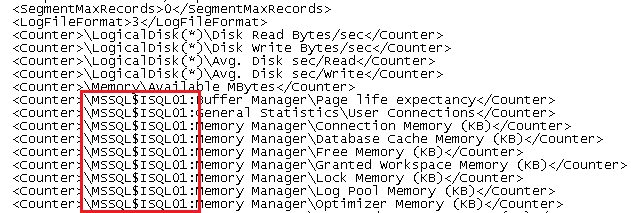
The XML template file is located at \\na01\Special\_projects\SQL\_Server\_Admin\ 04\_ScriptLibrary\ SQL\_PerfMonCollector20xx.XML

The XML may need to be altered to match the instance name of the SQL Server that requires monitoring. To check, open the XML file in notepad and check one of the SQL counters. If the instance to be monitored is a named instance, change the value

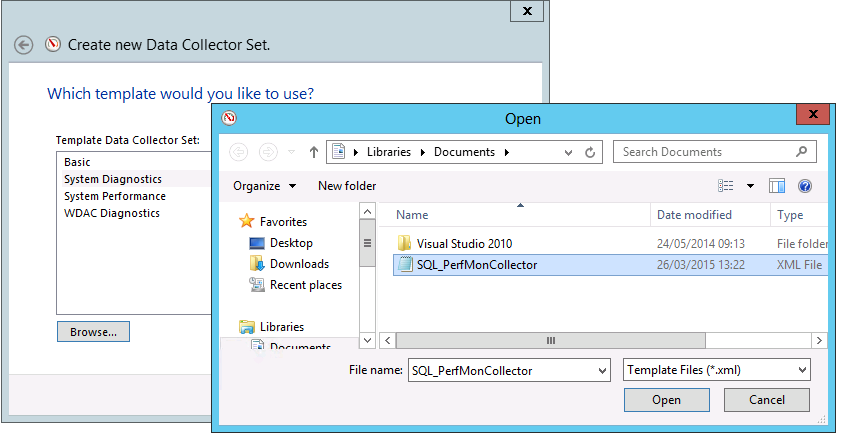
MSSQL$ISQL01 to MSSQL$<InstanceName>

If the instance is a default instance, change the value

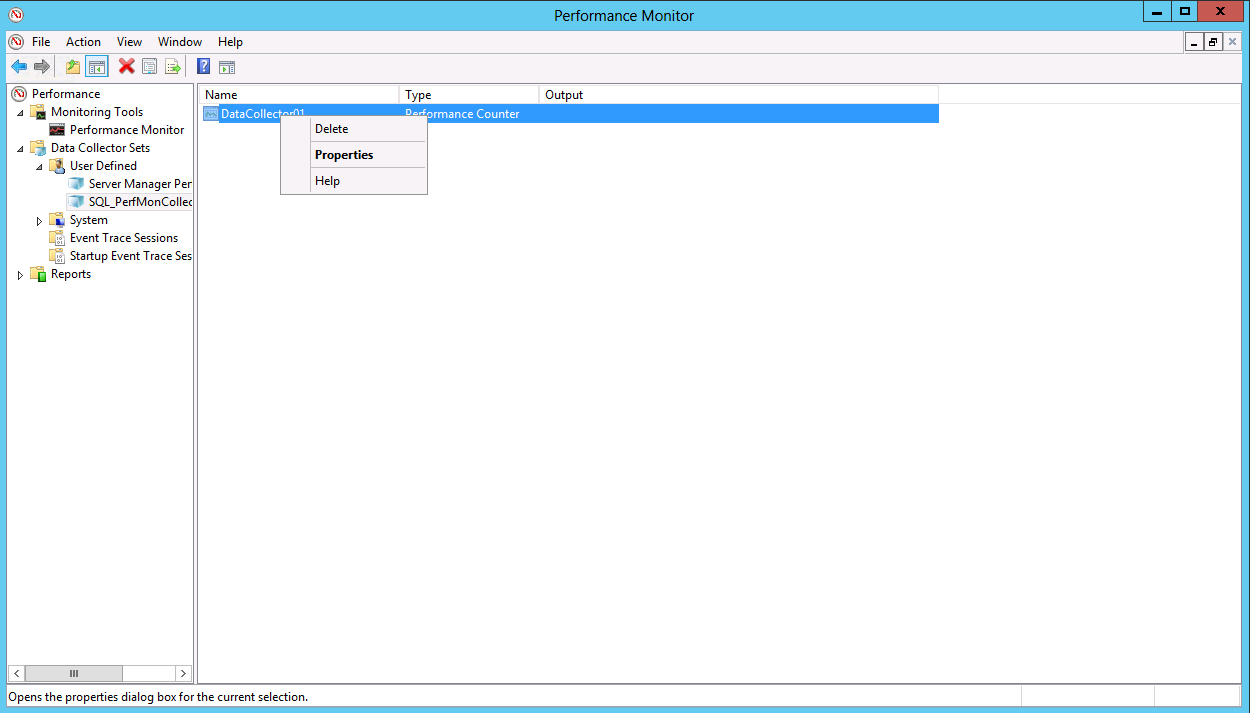
MSSQL$ISQL01 to SQLServer



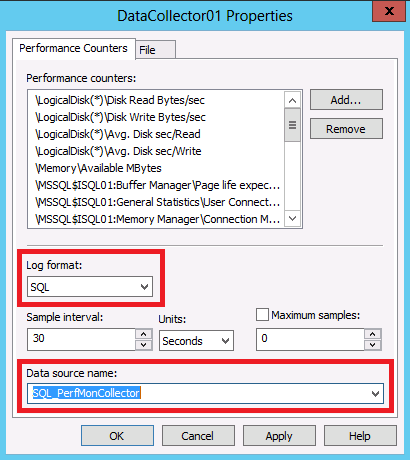
Browse to the location where the SQL\_PerfMonCollector is accessible and import. Choose defaults in next two screens and finish (with the save and close option).



Choose SQL\_PerfMonCollector under User Defined and then the properties of DataCollector01.



Change the Log Format to SQL and data source name to the name given in step 5 (SQL\_PerfMonCollector)



For instances on failover clusters where there are more than one instance contained on the cluster, the data collector set needs to be created on each node. The name of data collector set needs to be consistent across all nodes in the cluster and also unique to the instance.

### 7 – Add NonClustered Index and Page Compression

USE [zzPerfMon]

GO

CREATE NONCLUSTERED INDEX NIX\_CounterID

ON [dbo].[CounterData] ([CounterID])

INCLUDE ([CounterDateTime],[CounterValue]) WITH

(PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF,

SORT\_IN\_TEMPDB = ON, ONLINE = OFF,

ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON,

DATA\_COMPRESSION = PAGE)

USE [zzPerfMon]

GO

ALTER INDEX <IndexName> ON [dbo].[CounterData]

REBUILD PARTITION = ALL WITH

(PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF,

SORT\_IN\_TEMPDB = ON, ONLINE = OFF,

ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON,

DATA\_COMPRESSION = PAGE)

### 8 – Remove SQL\_PerfMonCollector

Remote onto server using an account with Local Administrator rights and remove SQL\_PerfMonCollector from the Administrators group.

### 9 – Create SQL Agent Job

Using an account with sysadmin rights, create SQL Agent Job. The job checks every hour if the PerfMon counter is running by executing a Powershell script. The job also purges data from the database for data older than 180 days (by default).

USE [msdb]

GO

/\*\*\*\*\*\* Object: Job [P1\_zzPerfMonMaintJobs] Script Date: 21/07/2015 10:09:45 \*\*\*\*\*\*/

BEGIN TRANSACTION

DECLARE @ReturnCode INT

SELECT @ReturnCode = 0

/\*\*\*\*\*\* Object: JobCategory [Database Maintenance] Script Date: 21/07/2015 10:09:45 \*\*\*\*\*\*/

IF NOT EXISTS (SELECT name FROM msdb.dbo.syscategories WHERE name=N'Database Maintenance' AND category\_class=1)

BEGIN

EXEC @ReturnCode = msdb.dbo.sp\_add\_category @class=N'JOB', @type=N'LOCAL', @name=N'Database Maintenance'

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

END

DECLARE @jobId BINARY(16)

EXEC @ReturnCode = msdb.dbo.sp\_add\_job @job\_name=N'P1\_zzPerfMonMaintJobs',

@enabled=1,

@notify\_level\_eventlog=0,

@notify\_level\_email=2,

@notify\_level\_netsend=0,

@notify\_level\_page=0,

@delete\_level=0,

@description=N'No description available.',

@category\_name=N'Database Maintenance',

@owner\_login\_name=N'sa',

@notify\_email\_operator\_name=N'SQLInsightServices', @job\_id = @jobId OUTPUT

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

/\*\*\*\*\*\* Object: Step [Check Data Collector] Script Date: 21/07/2015 10:09:45 \*\*\*\*\*\*/

EXEC @ReturnCode = msdb.dbo.sp\_add\_jobstep @job\_id=@jobId, @step\_name=N'Check Data Collector',

@step\_id=1,

@cmdexec\_success\_code=0,

@on\_success\_action=3,

@on\_success\_step\_id=0,

@on\_fail\_action=2,

@on\_fail\_step\_id=0,

@retry\_attempts=0,

@retry\_interval=0,

@os\_run\_priority=0, @subsystem=N'PowerShell',

@command=N'$datacollectorset = New-Object -COM Pla.DataCollectorSet;

$datacollectorset.Query("SQL\_PerfMonCollector",$null)

$status = $datacollectorset.Status

if ($status -eq 0) {$datacollectorset.start($false)}',

@database\_name=N'master',

@flags=0

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

/\*\*\*\*\*\* Object: Step [spPurgeCounterData] Script Date: 21/07/2015 10:09:45 \*\*\*\*\*\*/

EXEC @ReturnCode = msdb.dbo.sp\_add\_jobstep @job\_id=@jobId, @step\_name=N'spPurgeCounterData',

@step\_id=2,

@cmdexec\_success\_code=0,

@on\_success\_action=3,

@on\_success\_step\_id=0,

@on\_fail\_action=2,

@on\_fail\_step\_id=0,

@retry\_attempts=0,

@retry\_interval=0,

@os\_run\_priority=0, @subsystem=N'TSQL',

@command=N'EXEC dbo.spPurgeCounterData',

@database\_name=N'zzPerfMon',

@flags=0

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

/\*\*\*\*\*\* Object: Step [Alert Check] Script Date: 21/07/2015 10:09:45 \*\*\*\*\*\*/

EXEC @ReturnCode = msdb.dbo.sp\_add\_jobstep @job\_id=@jobId, @step\_name=N'Alert Check',

@step\_id=3,

@cmdexec\_success\_code=0,

@on\_success\_action=1,

@on\_success\_step\_id=0,

@on\_fail\_action=2,

@on\_fail\_step\_id=0,

@retry\_attempts=0,

@retry\_interval=0,

@os\_run\_priority=0, @subsystem=N'TSQL',

@command=N'

DECLARE @SQL1 nvarchar(1024) = (SELECT TOP 1 LogStopTime FROM dbo.DisplayToID WITH (NOLOCK) ORDER BY 1 DESC)

DECLARE @SQL2 nvarchar(1024) = ''SELECT 1 FROM sys.dm\_tran\_locks l INNER JOIN sys.dm\_exec\_sessions s ON l.request\_Session\_Id = s.session\_Id INNER JOIN sys.tables t ON l.resource\_associated\_entity\_id = t.object\_id WHERE db\_Name(l.resource\_database\_id) = ''''zzPerfMon'''' AND DATEDIFF(SS,s.last\_request\_end\_time,CURRENT\_TIMESTAMP) > 10 AND t.name = ''''DisplayToID''''''

CREATE TABLE #LockTest (LockTest int)

DECLARE @profile\_name sysname = (SELECT value FROM zzSQLServerAdmin.dbo.tblConfigs WHERE name = ''profile\_name'')

DECLARE @recipients varchar(max) = (SELECT value FROM zzSQLServerAdmin.dbo.tblConfigs WHERE name = ''recipients'')

DECLARE @subject1 nvarchar(255) = ''zzPerfMon collection delay on Instance = '' + @@SERVERNAME + ''''

DECLARE @subject2 nvarchar(255) = ''zzPerfMon collection long running update on Instance = '' + @@SERVERNAME + ''''

DECLARE @body nvarchar(1024) = ''REMOTE ONTO THE SERVER AND RUN THE FOLLOWING

--in powershell

$datacollectorset = New-Object -COM Pla.DataCollectorSet;

$datacollectorset.Query("SQL\_PerfMonCollector",$null)

$datacollectorset.stop($false)

SQLCMD -E -Slocalhost

USE zzPerfMon

GO

UPDATE dbo.DisplayToID SET NumberOfRecords = (SELECT MAX(RecordIndex) FROM dbo.CounterData)

GO

EXIT

$datacollectorset = New-Object -COM Pla.DataCollectorSet;

$datacollectorset.Query("SQL\_PerfMonCollector",$null)

$datacollectorset.start($false)

''

INSERT INTO #LockTest (LockTest)

EXEC(@SQL2)

WAITFOR DELAY ''00:00:10''

INSERT INTO #LockTest (LockTest)

EXEC(@SQL2)

IF @SQL1 < DATEADD(DD, -1, CURRENT\_TIMESTAMP)

BEGIN

EXEC [zzSQLServerAdmin].dbo.[spSendDBMail] @profile\_name,@recipients,@subject1

END

IF EXISTS(SELECT 1 FROM #LockTest)

BEGIN

EXEC [zzSQLServerAdmin].dbo.[spSendDBMail] @profile\_name,@recipients,@subject2, @body

END

',

@database\_name=N'zzPerfMon',

@flags=0

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

EXEC @ReturnCode = msdb.dbo.sp\_update\_job @job\_id = @jobId, @start\_step\_id = 1

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

EXEC @ReturnCode = msdb.dbo.sp\_add\_jobschedule @job\_id=@jobId, @name=N'Every Hour',

@enabled=1,

@freq\_type=4,

@freq\_interval=1,

@freq\_subday\_type=8,

@freq\_subday\_interval=1,

@freq\_relative\_interval=0,

@freq\_recurrence\_factor=0,

@active\_start\_date=20140423,

@active\_end\_date=99991231,

@active\_start\_time=0,

@active\_end\_time=235959,

@schedule\_uid=N'c5107d5a-4d40-4bdb-8c15-8c212db6d22e'

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

EXEC @ReturnCode = msdb.dbo.sp\_add\_jobschedule @job\_id=@jobId, @name=N'Start Up',

@enabled=1,

@freq\_type=64,

@freq\_interval=0,

@freq\_subday\_type=0,

@freq\_subday\_interval=0,

@freq\_relative\_interval=0,

@freq\_recurrence\_factor=0,

@active\_start\_date=20140423,

@active\_end\_date=99991231,

@active\_start\_time=0,

@active\_end\_time=235959,

@schedule\_uid=N'7655e502-9538-465d-884c-17a6ec371578'

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

EXEC @ReturnCode = msdb.dbo.sp\_add\_jobserver @job\_id = @jobId, @server\_name = N'(local)'

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

COMMIT TRANSACTION

GOTO EndSave

QuitWithRollback:

IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION

EndSave:

GO

### 10 – Purge Data

Create stored procedure to purge data from database at desired interval.

USE zzPerfMon

GO

CREATE PROCEDURE dbo.spPurgeCounterData @DaysToKeep INT = 180

AS

DELETE

FROM [zzPerfMon].[dbo].[CounterData]

WHERE CAST(SUBSTRING(CounterDateTime,1,19) AS SMALLDATETIME) < CURRENT\_TIMESTAMP - @DaysToKeep

### 11 – Create Views and Functions

Create Views to query data (copy and paste all TSQL below and execute in one go. Must delete and rename views and functions with wrong edition)

USE zzPerfMon

Go

CREATE View [dbo].[vwCounterData]

AS

SELECT

[GUID],

[CounterID],

[RecordIndex],

[CounterDateTime],

[CounterValue],

[FirstValueA],

[FirstValueB],

[SecondValueA],

[SecondValueB],

[MultiCount],

CONVERT([datetime],substring([CounterDateTime],(0),charindex('.',[CounterDateTime])),(20)) as [CounterDateTimeFormat]

FROM dbo.CounterData

GO

USE zzPerfMon

Go

Create View vwCPU

As

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2) AS Date

, CAST(AVG(CounterValue) AS INT) as AvgCPU

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.countername = '% Processor Time' AND cd.InstanceName = '\_Total'

GROUP BY SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2)

GO

USE zzPerfMon

GO

Create View vwPageLifeExpectancy

As

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2) AS Date

,CAST(AVG(CounterValue) AS INT) as AvgPageLifeExpectancy

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.countername = 'Page life expectancy'

GROUP BY SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2)

GO

USE zzPerfMon

GO

Create view vwAvailableMemoryMB

AS

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2) AS Date

, CAST(AVG(CounterValue) AS INT) as AvgAvailableMemoryMB

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.countername = 'Available MBytes'

GROUP BY SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2)

GO

USE zzPerfMon

GO

Create view vwUserConnections

AS

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2) AS Date

,CAST(AVG(CounterValue) AS INT) as AvgUserConnections

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.countername = 'User Connections'

GROUP BY SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2)

GO

USE zzPerfMon

GO

Create view vwBatchRequestsSec

AS

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2) AS Date

,CAST(AVG(CounterValue) AS INT) as AvgBatchRequestsSec

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.countername = 'Batch Requests/sec'

GROUP BY SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2)

GO

USE zzPerfMon

GO

CREATE VIEW [dbo].[vwBufferCacheMemoryMB]

AS

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2) AS Date

,CAST(AVG(CounterValue)/1024 AS INT) as BufferCacheSizeMB

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.CounterName = ('Database Cache Memory (KB)')

GROUP BY SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2)

UNION

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2) AS Date

,CAST(AVG(CounterValue\*8)/1024 AS INT) as BufferCacheSizeMB

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.CounterName = 'Database pages'

GROUP BY SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2)

GO

USE zzPerfMon

GO

Create View vwMemory\_2012

AS

SELECT Date,

ROUND([Total Server Memory (KB)]/1024, 0) as 'Total Server Memory (MB)'

,ROUND([Target Server Memory (KB)]/1024 ,0) as 'Target Server Memory (MB)'

,ROUND([Database Cache Memory (KB)]/1024 ,0) AS 'Database Cache Memory (MB)'

,ROUND([Stolen Server Memory (KB)]/1024 ,0) as 'Stolen Server Memory (MB)'

,ROUND([Granted Workspace Memory (KB)]/1024 ,0) as 'Granted Workspace Memory (MB)'

,ROUND([Free Memory (KB)]/1024, 0) as 'Free Memory (MB)'

,ROUND([SQL Cache Memory (KB)]/1024 ,0) as 'SQL Cache Memory (MB)'

,ROUND([Optimizer Memory (KB)]/1024 ,0) as 'Optimizer Memory (MB)'

,ROUND([Log Pool Memory (KB)]/1024 ,0) as 'Log Pool Memory (MB)'

,ROUND([Lock Memory (KB)]/1024 ,0) as 'Lock Memory (MB)'

,ROUND([Connection Memory (KB)]/1024 ,0) as 'Connection Memory (MB)'

FROM

(

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2) AS Date

,cd.countername

,CounterValue as AvgCounterValue

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.countername IN (

'Database Cache Memory (KB)','Total Server Memory (KB)','Target Server Memory (KB)','Stolen Server Memory (KB)', 'SQL Cache Memory (KB)', 'Optimizer Memory (KB)', 'Log Pool Memory (KB)','Lock Memory (KB)','Granted Workspace Memory (KB)','Free Memory (KB)','Connection Memory (KB)')

) src

pivot

(

AVG(AvgCounterValue)

for countername in

([Database Cache Memory (KB)], [Total Server Memory (KB)], [Target Server Memory (KB)], [Stolen Server Memory (KB)], [SQL Cache Memory (KB)]

, [Optimizer Memory (KB)], [Log Pool Memory (KB)], [Lock Memory (KB)], [Granted Workspace Memory (KB)], [Free Memory (KB)], [Connection Memory (KB)]

)

) piv;

GO

USE zzPerfMon

GO

CREATE VIEW vwMemory\_2008

AS

SELECT Date,

ROUND([Total Server Memory (KB)]/1024, 0) as 'Total Server Memory (MB)'

,ROUND([Target Server Memory (KB)]/1024 ,0) as 'Target Server Memory (MB)'

,ROUND([Database pages]\*8/1024 ,0) AS 'Database Cache Memory (MB)'

,ROUND([Stolen pages]\*8/1024 ,0) as 'Stolen Server Memory (MB)'

,ROUND([Granted Workspace Memory (KB)]/1024 ,0) as 'Granted Workspace Memory (MB)'

,ROUND([Free pages]\*8/1024, 0) as 'Free Memory (MB)'

,ROUND([SQL Cache Memory (KB)]/1024 ,0) as 'SQL Cache Memory (MB)'

,ROUND([Optimizer Memory (KB)]/1024 ,0) as 'Optimizer Memory (MB)'

,ROUND([Log Pool Memory (KB)]/1024 ,0) as 'Log Pool Memory (MB)'

,ROUND([Lock Memory (KB)]/1024 ,0) as 'Lock Memory (MB)'

,ROUND([Connection Memory (KB)]/1024 ,0) as 'Connection Memory (MB)'

FROM

(

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + '::' + SUBSTRING(CounterDateTime,12,2) AS Date

,cd.countername

, CounterValue as AvgCounterValue

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.countername IN (

'Database pages','Total Server Memory (KB)','Target Server Memory (KB)','Stolen pages', 'SQL Cache Memory (KB)', 'Optimizer Memory (KB)', 'Log Pool Memory (KB)','Lock Memory (KB)','Granted Workspace Memory (KB)','Free pages','Connection Memory (KB)')

) src

pivot

(

AVG(AvgCounterValue)

for countername in

([Database pages], [Total Server Memory (KB)], [Target Server Memory (KB)], [Stolen pages], [SQL Cache Memory (KB)]

, [Optimizer Memory (KB)], [Log Pool Memory (KB)], [Lock Memory (KB)], [Granted Workspace Memory (KB)], [Free pages], [Connection Memory (KB)]

)

) piv;

GO

USE [zzPerfMon]

GO

DECLARE @SQL AS NVARCHAR(MAX)

DECLARE @ColumnName AS NVARCHAR(MAX)

DECLARE @ColumnName1 AS NVARCHAR(MAX)

--Get distinct values of the PIVOT Column

SELECT @ColumnName = ISNULL(@ColumnName + ',','') + QUOTENAME(InstanceName) + ' = ROUND(' + QUOTENAME(InstanceName) + '\*30/1024/1024 ,0)'

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Read Bytes/sec','Disk Write Bytes/sec')) AS Disks

SELECT @ColumnName1 = ISNULL(@ColumnName1 + ',','') + QUOTENAME(InstanceName)

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Read Bytes/sec','Disk Write Bytes/sec')) AS Disks

SET @SQL =

'

CREATE VIEW vwIO\_TotalDiskMB

AS

SELECT Date,CounterName =

CASE countername

WHEN ''Disk Read Bytes/sec'' THEN ''Total Disk Read MB''

WHEN ''Disk Write Bytes/sec'' THEN ''Total Disk Write MB''

END

,' + @ColumnName + '

From

(

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + ''::'' + SUBSTRING(CounterDateTime,12,2) AS Date

,cd.countername, cd.InstanceName

,CounterValue as AvgCounterValue

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.countername IN (

''Disk Read Bytes/sec'',''Disk Write Bytes/sec'')

) src

Pivot

(

SUM(AvgCounterValue)

For InstanceName in

(

' + @ColumnName1 + '

)

) piv'

EXEC(@SQL)

GO

USE [zzPerfMon]

GO

DECLARE @SQL AS NVARCHAR(MAX)

DECLARE @ColumnName AS NVARCHAR(MAX)

DECLARE @ColumnName1 AS NVARCHAR(MAX)

--Get distinct values of the PIVOT Column

SELECT @ColumnName = ISNULL(@ColumnName + ',','') + QUOTENAME(InstanceName) + ' = ROUND(' + QUOTENAME(InstanceName) + '/1024/1024 ,3)'

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Read Bytes/sec','Disk Write Bytes/sec')) AS Disks

SELECT @ColumnName1 = ISNULL(@ColumnName1 + ',','') + QUOTENAME(InstanceName)

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Read Bytes/sec','Disk Write Bytes/sec')) AS Disks

SET @SQL =

'

CREATE VIEW vwIO\_AvgDiskMBSec

AS

SELECT Date,CounterName =

CASE countername

WHEN ''Disk Read Bytes/sec'' THEN ''Disk Read MB/sec''

WHEN ''Disk Write Bytes/sec'' THEN ''Disk Write MB/sec''

END

,' + @ColumnName + '

from

(

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + ''::'' + SUBSTRING(CounterDateTime,12,2) AS Date

,cd.countername, cd.InstanceName

,CounterValue as AvgCounterValue

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.countername IN (

''Disk Read Bytes/sec'',''Disk Write Bytes/sec'')

) src

pivot

(

AVG(AvgCounterValue)

for InstanceName in

(

' + @ColumnName1 + '

)

) piv'

EXEC(@SQL)

GO

USE [zzPerfMon]

GO

DECLARE @SQL AS NVARCHAR(MAX)

DECLARE @ColumnName AS NVARCHAR(MAX)

DECLARE @ColumnName1 AS NVARCHAR(MAX)

--Get distinct values of the PIVOT Column

SELECT @ColumnName = ISNULL(@ColumnName + ',','') + QUOTENAME(InstanceName) + ' = ROUND(' + QUOTENAME(InstanceName) + '\*1000 ,0)'

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Read Bytes/sec','Disk Write Bytes/sec')) AS Disks

SELECT @ColumnName1 = ISNULL(@ColumnName1 + ',','') + QUOTENAME(InstanceName)

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Read Bytes/sec','Disk Write Bytes/sec')) AS Disks

SET @SQL =

'

CREATE VIEW vwIO\_AvgDiskMS

AS

SELECT Date,CounterName =

CASE countername

WHEN ''Avg. Disk sec/Read'' THEN ''Avg. Disk MS/Read''

WHEN ''Avg. Disk sec/WRITE'' THEN ''Avg. Disk MS/Write''

END

,' + @ColumnName + '

from

(

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + ''::'' + SUBSTRING(CounterDateTime,12,2) AS Date

,cd.countername, cd.InstanceName

,CounterValue as AvgCounterValue

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.countername IN (

''Avg. Disk sec/Read'',''Avg. Disk sec/Write'')

) src

Pivot

(

AVG(AvgCounterValue)

for InstanceName in

(

' + @ColumnName1 + '

)

) piv'

EXEC(@SQL)

GO

USE [zzPerfMon]

GO

DECLARE @SQL AS NVARCHAR(MAX)

DECLARE @ColumnName AS NVARCHAR(MAX)

DECLARE @ColumnName1 AS NVARCHAR(MAX)

--Get distinct values of the PIVOT Column

SELECT @ColumnName = ISNULL(@ColumnName + ',','') + QUOTENAME(InstanceName) + ' = ROUND(' + QUOTENAME(InstanceName) + '\*30,0)'

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Reads/sec')) AS Disks

SELECT @ColumnName1 = ISNULL(@ColumnName1 + ',','') + QUOTENAME(InstanceName)

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Reads/sec')) AS Disks

SET @SQL =

'

CREATE VIEW vwIO\_AvgReadsSec

AS

SELECT Date,CounterName

,' + @ColumnName + '

From

(

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + ''::'' + SUBSTRING(CounterDateTime,12,2) AS Date

,cd.countername, cd.InstanceName

,CounterValue as AvgCounterValue

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.countername IN (

''Disk Reads/sec'')

) src

Pivot

(

AVG(AvgCounterValue)

For InstanceName in

(

' + @ColumnName1 + '

)

) piv'

EXEC(@SQL)

GO

USE [zzPerfMon]

GO

DECLARE @SQL AS NVARCHAR(MAX)

DECLARE @ColumnName AS NVARCHAR(MAX)

DECLARE @ColumnName1 AS NVARCHAR(MAX)

--Get distinct values of the PIVOT Column

SELECT @ColumnName = ISNULL(@ColumnName + ',','') + QUOTENAME(InstanceName) + ' = ROUND(' + QUOTENAME(InstanceName) + '\*30,0)'

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Writes/sec')) AS Disks

SELECT @ColumnName1 = ISNULL(@ColumnName1 + ',','') + QUOTENAME(InstanceName)

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Writes/sec')) AS Disks

SET @SQL =

'

CREATE VIEW vwIO\_AvgWritesSec

AS

SELECT Date,CounterName

,' + @ColumnName + '

From

(

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + ''::'' + SUBSTRING(CounterDateTime,12,2) AS Date

,cd.countername, cd.InstanceName

,CounterValue as AvgCounterValue

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.countername IN (

''Disk Writes/sec'')

) src

Pivot

(

Avg(AvgCounterValue)

For InstanceName in

(

' + @ColumnName1 + '

)

) piv'

EXEC(@SQL)

GO

USE [zzPerfMon]

GO

DECLARE @SQL AS NVARCHAR(MAX)

DECLARE @ColumnName AS NVARCHAR(MAX)

DECLARE @ColumnName1 AS NVARCHAR(MAX)

--Get distinct values of the PIVOT Column

SELECT @ColumnName = ISNULL(@ColumnName + ',','') + QUOTENAME(InstanceName) + ' = ROUND(' + QUOTENAME(InstanceName) + ',0)'

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Current Disk Queue Length')) AS Disks

SELECT @ColumnName1 = ISNULL(@ColumnName1 + ',','') + QUOTENAME(InstanceName)

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Current Disk Queue Length')) AS Disks

SET @SQL =

'

CREATE VIEW vwIO\_AvgCurrentDiskQueueLength

AS

SELECT Date,CounterName

,' + @ColumnName + '

From

(

SELECT SUBSTRING(CounterDateTime,1,4) + SUBSTRING(CounterDateTime,6,2) + SUBSTRING(CounterDateTime,9,2) + ''::'' + SUBSTRING(CounterDateTime,12,2) AS Date

,cd.countername, cd.InstanceName

,CounterValue as AvgCounterValue

FROM dbo.CounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE cd.countername IN (

''Current Disk Queue Length'')

) src

Pivot

(

avg(AvgCounterValue)

For InstanceName in

(

' + @ColumnName1 + '

)

) piv'

EXEC(@SQL)

GO

USE [zzPerfMon]

GO

DECLARE @SQL AS NVARCHAR(MAX)

DECLARE @ColumnName AS NVARCHAR(MAX)

DECLARE @ColumnName1 AS NVARCHAR(MAX)

--Get distinct values of the PIVOT Column

SELECT @ColumnName = ISNULL(@ColumnName + ',','') + QUOTENAME(InstanceName) + ' = ROUND(' + QUOTENAME(InstanceName) + ',0)'

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Current Disk Queue Length')) AS Disks

SELECT @ColumnName1 = ISNULL(@ColumnName1 + ',','') + QUOTENAME(InstanceName)

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Current Disk Queue Length')) AS Disks

SET @SQL =

'

CREATE FUNCTION fnIO\_CurrentDiskQueueLength

(

@StartTime DATETIME,

@EndTime DATETIME

)

RETURNS TABLE

AS

RETURN (

SELECT [CounterDateTimeFormat],CounterName

,' + @ColumnName + '

From

(

SELECT [CounterDateTimeFormat]

,cd.countername, cd.InstanceName

,CounterValue as AvgCounterValue

FROM dbo.vwCounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE

d.[CounterDateTimeFormat] > @StartTime AND d.[CounterDateTimeFormat] < @EndTime AND

cd.countername IN (''Current Disk Queue Length'')

) src

Pivot

(

avg(AvgCounterValue)

For InstanceName in

(

' + @ColumnName1 + '

)

) piv

)

'

EXEC(@SQL)

GO

USE [zzPerfMon]

GO

DECLARE @SQL AS NVARCHAR(MAX)

DECLARE @ColumnName AS NVARCHAR(MAX)

DECLARE @ColumnName1 AS NVARCHAR(MAX)

--Get distinct values of the PIVOT Column

SELECT @ColumnName = ISNULL(@ColumnName + ',','') + QUOTENAME(InstanceName) + ' = ROUND(' + QUOTENAME(InstanceName) + ',0)'

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Reads/Sec')) AS Disks

SELECT @ColumnName1 = ISNULL(@ColumnName1 + ',','') + QUOTENAME(InstanceName)

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Reads/Sec')) AS Disks

SET @SQL =

'

CREATE FUNCTION fnIO\_DiskReadsSec

(

@StartTime DATETIME,

@EndTime DATETIME

)

RETURNS TABLE

AS

RETURN (

SELECT [CounterDateTimeFormat],CounterName

,' + @ColumnName + '

From

(

SELECT [CounterDateTimeFormat]

,cd.countername, cd.InstanceName

,CounterValue as AvgCounterValue

FROM dbo.vwCounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE

d.[CounterDateTimeFormat] > @StartTime AND d.[CounterDateTimeFormat] < @EndTime AND

cd.countername IN (''Disk Reads/Sec'')

) src

Pivot

(

avg(AvgCounterValue)

For InstanceName in

(

' + @ColumnName1 + '

)

) piv

)

'

EXEC(@SQL)

GO

USE [zzPerfMon]

GO

DECLARE @SQL AS NVARCHAR(MAX)

DECLARE @ColumnName AS NVARCHAR(MAX)

DECLARE @ColumnName1 AS NVARCHAR(MAX)

--Get distinct values of the PIVOT Column

SELECT @ColumnName = ISNULL(@ColumnName + ',','') + QUOTENAME(InstanceName) + ' = ROUND(' + QUOTENAME(InstanceName) + ',0)'

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Writes/Sec')) AS Disks

SELECT @ColumnName1 = ISNULL(@ColumnName1 + ',','') + QUOTENAME(InstanceName)

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Disk Writes/Sec')) AS Disks

SET @SQL =

'

CREATE FUNCTION fnIO\_DiskWritesSec

(

@StartTime DATETIME,

@EndTime DATETIME

)

RETURNS TABLE

AS

RETURN (

SELECT [CounterDateTimeFormat],CounterName

,' + @ColumnName + '

From

(

SELECT [CounterDateTimeFormat]

,cd.countername, cd.InstanceName

,CounterValue as AvgCounterValue

FROM dbo.vwCounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE

d.[CounterDateTimeFormat] > @StartTime AND d.[CounterDateTimeFormat] < @EndTime AND

cd.countername IN (''Disk Writes/Sec'')

) src

Pivot

(

avg(AvgCounterValue)

For InstanceName in

(

' + @ColumnName1 + '

)

) piv

)

'

EXEC(@SQL)

GO

USE [zzPerfMon]

GO

DECLARE @SQL AS NVARCHAR(MAX)

DECLARE @ColumnName AS NVARCHAR(MAX)

DECLARE @ColumnName1 AS NVARCHAR(MAX)

SELECT @ColumnName = ISNULL(@ColumnName + ',','') + QUOTENAME(InstanceName) + ' = ROUND(' + QUOTENAME(InstanceName) + '\*1000,0)'

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Avg. Disk sec/Read')) AS Disks

SELECT @ColumnName1 = ISNULL(@ColumnName1 + ',','') + QUOTENAME(InstanceName)

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Avg. Disk sec/Read')) AS Disks

SET @SQL =

'

CREATE FUNCTION fnIO\_AvgDiskSecRead

(

@StartTime DATETIME,

@EndTime DATETIME

)

RETURNS TABLE

AS

RETURN (

SELECT [CounterDateTimeFormat],CounterName

,' + @ColumnName + '

From

(

SELECT [CounterDateTimeFormat]

,cd.countername, cd.InstanceName

,CounterValue as AvgCounterValue

FROM dbo.vwCounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE

d.[CounterDateTimeFormat] > @StartTime AND d.[CounterDateTimeFormat] < @EndTime AND

cd.countername IN (''Avg. Disk sec/Read'')

) src

Pivot

(

avg(AvgCounterValue)

For InstanceName in

(

' + @ColumnName1 + '

)

) piv

)

'

EXEC(@SQL)

GO

USE [zzPerfMon]

GO

DECLARE @SQL AS NVARCHAR(MAX)

DECLARE @ColumnName AS NVARCHAR(MAX)

DECLARE @ColumnName1 AS NVARCHAR(MAX)

SELECT @ColumnName = ISNULL(@ColumnName + ',','') + QUOTENAME(InstanceName) + ' = ROUND(' + QUOTENAME(InstanceName) + '\*1000,0)'

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Avg. Disk sec/Write')) AS Disks

SELECT @ColumnName1 = ISNULL(@ColumnName1 + ',','') + QUOTENAME(InstanceName)

FROM (SELECT DISTINCT InstanceName FROM dbo.CounterDetails WHERE CounterName IN ('Avg. Disk sec/Write')) AS Disks

SET @SQL =

'

CREATE FUNCTION fnIO\_AvgDiskSecWrite

(

@StartTime DATETIME,

@EndTime DATETIME

)

RETURNS TABLE

AS

RETURN (

SELECT [CounterDateTimeFormat],CounterName

,' + @ColumnName + '

From

(

SELECT [CounterDateTimeFormat]

,cd.countername, cd.InstanceName

,CounterValue as AvgCounterValue

FROM dbo.vwCounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE

d.[CounterDateTimeFormat] > @StartTime AND d.[CounterDateTimeFormat] < @EndTime AND

cd.countername IN (''Avg. Disk sec/Write'')

) src

Pivot

(

avg(AvgCounterValue)

For InstanceName in

(

' + @ColumnName1 + '

)

) piv

)

'

EXEC(@SQL)

GO

USE [zzPerfMon]

GO

CREATE FUNCTION [dbo].[fnAvailableMemoryMB]

(

@StartTime DATETIME,

@EndTime DATETIME

)

RETURNS TABLE

AS

RETURN

(

SELECT CounterDateTimeFormat

, CounterValue AS AvailableMemoryMB

FROM dbo.vwCounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE

d.[CounterDateTimeFormat] > @StartTime AND d.[CounterDateTimeFormat] < @EndTime AND

cd.countername = 'Available MBytes'

)

GO

USE [zzPerfMon]

GO

CREATE FUNCTION [dbo].[fnBatchRequestsSec]

(

@StartTime DATETIME,

@EndTime DATETIME

)

RETURNS TABLE

AS

RETURN

(

SELECT CounterDateTimeFormat

, cast(CounterValue as int) AS BatchRequestsSec

FROM dbo.vwCounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE

d.[CounterDateTimeFormat] > @StartTime AND d.[CounterDateTimeFormat] < @EndTime AND

cd.countername = 'Batch Requests/sec'

)

GO

USE [zzPerfMon]

GO

CREATE FUNCTION [dbo].[fnCPU]

(

@StartTime DATETIME,

@EndTime DATETIME

)

RETURNS TABLE

AS

RETURN

(

SELECT CounterDateTimeFormat

, CAST(CounterValue AS INT) AS CPU

FROM dbo.vwCounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE

d.[CounterDateTimeFormat] > @StartTime AND d.[CounterDateTimeFormat] < @EndTime AND

cd.countername = '% Processor Time'

)

GO

USE [zzPerfMon]

GO

CREATE FUNCTION [dbo].[fnPageLifeExpectancy]

(

@StartTime DATETIME,

@EndTime DATETIME

)

RETURNS TABLE

AS

RETURN

(

SELECT CounterDateTimeFormat

, CounterValue AS PageLifeExpectancy

FROM dbo.vwCounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE

d.[CounterDateTimeFormat] > @StartTime AND d.[CounterDateTimeFormat] < @EndTime AND

cd.countername = 'Page life expectancy'

)

GO

USE [zzPerfMon]

GO

CREATE FUNCTION [dbo].[fnUserConnections]

(

@StartTime DATETIME,

@EndTime DATETIME

)

RETURNS TABLE

AS

RETURN

(

SELECT CounterDateTimeFormat

, CounterValue AS UserConnections

FROM dbo.vwCounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE

d.[CounterDateTimeFormat] > @StartTime AND d.[CounterDateTimeFormat] < @EndTime AND

cd.countername = 'User Connections'

)

GO

USE zzPerfMon

GO

CREATE FUNCTION fnMemory\_2012

(

@StartTime DATETIME,

@EndTime DATETIME

)

RETURNS TABLE

AS

RETURN

(

SELECT [CounterDateTimeFormat],

ROUND([Total Server Memory (KB)]/1024, 0) as 'Total Server Memory (MB)'

,ROUND([Target Server Memory (KB)]/1024 ,0) as 'Target Server Memory (MB)'

,ROUND([Database Cache Memory (KB)]/1024 ,0) AS 'Database Cache Memory (MB)'

,ROUND([Stolen Server Memory (KB)]/1024 ,0) as 'Stolen Server Memory (MB)'

,ROUND([Granted Workspace Memory (KB)]/1024 ,0) as 'Granted Workspace Memory (MB)'

,ROUND([Free Memory (KB)]/1024, 0) as 'Free Memory (MB)'

,ROUND([SQL Cache Memory (KB)]/1024 ,0) as 'SQL Cache Memory (MB)'

,ROUND([Optimizer Memory (KB)]/1024 ,0) as 'Optimizer Memory (MB)'

,ROUND([Log Pool Memory (KB)]/1024 ,0) as 'Log Pool Memory (MB)'

,ROUND([Lock Memory (KB)]/1024 ,0) as 'Lock Memory (MB)'

,ROUND([Connection Memory (KB)]/1024 ,0) as 'Connection Memory (MB)'

FROM

(

SELECT d.[CounterDateTimeFormat],cd.countername

,CounterValue as AvgCounterValue

FROM dbo.vwCounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE

d.[CounterDateTimeFormat] > @StartTime AND d.[CounterDateTimeFormat] < @EndTime AND

cd.countername IN (

'Database Cache Memory (KB)','Total Server Memory (KB)','Target Server Memory (KB)','Stolen Server Memory (KB)', 'SQL Cache Memory (KB)', 'Optimizer Memory (KB)', 'Log Pool Memory (KB)','Lock Memory (KB)','Granted Workspace Memory (KB)','Free Memory (KB)','Connection Memory (KB)')

) src

pivot

(

AVG(AvgCounterValue)

for countername in

([Database Cache Memory (KB)], [Total Server Memory (KB)], [Target Server Memory (KB)], [Stolen Server Memory (KB)], [SQL Cache Memory (KB)]

, [Optimizer Memory (KB)], [Log Pool Memory (KB)], [Lock Memory (KB)], [Granted Workspace Memory (KB)], [Free Memory (KB)], [Connection Memory (KB)]

)

) piv

)

GO

USE zzPerfMon

GO

CREATE FUNCTION fnMemory\_2008

(

@StartTime DATETIME,

@EndTime DATETIME

)

RETURNS TABLE

AS

RETURN

(

SELECT [CounterDateTimeFormat],

ROUND([Total Server Memory (KB)]/1024, 0) as 'Total Server Memory (MB)'

,ROUND([Target Server Memory (KB)]/1024 ,0) as 'Target Server Memory (MB)'

,ROUND([Database pages]\*8/1024 ,0) AS 'Database Cache Memory (MB)'

,ROUND([Stolen pages]\*8/1024 ,0) as 'Stolen Server Memory (MB)'

,ROUND([Granted Workspace Memory (KB)]/1024 ,0) as 'Granted Workspace Memory (MB)'

,ROUND([Free pages]\*8/1024, 0) as 'Free Memory (MB)'

,ROUND([SQL Cache Memory (KB)]/1024 ,0) as 'SQL Cache Memory (MB)'

,ROUND([Optimizer Memory (KB)]/1024 ,0) as 'Optimizer Memory (MB)'

,ROUND([Log Pool Memory (KB)]/1024 ,0) as 'Log Pool Memory (MB)'

,ROUND([Lock Memory (KB)]/1024 ,0) as 'Lock Memory (MB)'

,ROUND([Connection Memory (KB)]/1024 ,0) as 'Connection Memory (MB)'

FROM

(

SELECT d.[CounterDateTimeFormat],cd.countername

, CounterValue as AvgCounterValue

FROM dbo.vwCounterData d

INNER JOIN dbo.CounterDetails cd

ON cd.CounterID = d.CounterID

WHERE

d.[CounterDateTimeFormat] > @StartTime AND d.[CounterDateTimeFormat] < @EndTime AND

cd.countername IN (

'Database pages','Total Server Memory (KB)','Target Server Memory (KB)','Stolen pages', 'SQL Cache Memory (KB)', 'Optimizer Memory (KB)', 'Log Pool Memory (KB)','Lock Memory (KB)','Granted Workspace Memory (KB)','Free pages','Connection Memory (KB)')

) src

pivot

(

AVG(AvgCounterValue)

for countername in

([Database pages], [Total Server Memory (KB)], [Target Server Memory (KB)], [Stolen pages], [SQL Cache Memory (KB)]

, [Optimizer Memory (KB)], [Log Pool Memory (KB)], [Lock Memory (KB)], [Granted Workspace Memory (KB)], [Free pages], [Connection Memory (KB)]

)

) piv

)

GO